#9

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	ALEX I. EYDELBERG	§	Group Art Unit:	2135
Serial No.:	09/465,600	§	Examiner:	Leynna A. Ha
Filed:	December 17, 1999	§		
For:	DYNAMICALLY LINKED BASIC INPUT/OUTPUT SYSTEM	§	Atty. Dkt. No.:	ITL.0304US (P7882)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

Applicant respectfully appeals from the final rejection mailed February 20, 2004.

I. REAL PARTY IN INTEREST

The real party in interest is the assignee Intel Corporation, the assignee of the present application by virtue of the assignment recorded at Reel/Frame 010463/0513.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

The application was originally filed with claims 1-30. Claims 1-30 are pending. Claims 1-30 are the subject of this appeal.

Date of Deposit: July 7, 2004
I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as **first class mail** with sufficient postage on the date indicated above and is addressed to the Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA ~~22313-1450.~~

Jennifer Juarez

IV. STATUS OF AMENDMENTS

All Amendments previously presented by the Applicant have been entered as of the date of this Appeal.

V. SUMMARY OF THE INVENTION

This invention relates generally to processor-based systems and particularly to basic input/output systems (BIOS) that implement more elaborate capabilities than traditional BIOS. *See* Specification, pp. 1-2.

Referring to Figure 1, a processor-based system 10 such as a personal computer may include a processor 12 coupled to an interface 14. The interface 14 may a bridge or a part of a chipset, as examples. The interface 14 may also be coupled to a bus 24 such as a Peripheral Component Interconnect (PCI) bus. The bus 24 may be coupled to an interface 26 which may also be part of a chipset and which may be implemented by a bridge as examples. Also coupled to the interface 26 is a hard disk drive 28 which may store basic input/output system (BIOS) module 30. A module is a file that need not meet any of the requirements normally associated with a Disk Operating System (DOS) file.

Coupled to a legacy bus 42 is a network interface (NIC) card 32 that allows the system 10 to connect to a server 36 over a network 34. For example, in one embodiment of the present invention, the connection to the network and the assignment of Internet Protocol (IP) addresses may be handled using Dynamic Host Configuration Protocol (DHCP). The server 36 may include a storage device 38 which also stores BIOS module 40.

Coupled to a legacy bus 42 is a BIOS read only memory (ROM) 50 that may store additional BIOS module 52. In some embodiments of the present invention, additional BIOS file storage devices may be coupled to the bus 42 through the interface 44. *See* Specification, pp. 3-

4. The BIOS, which normally would be stored before execution entirely on the BIOS ROM 50, may be distributed among a plurality of modules stored in a plurality of storage devices. Referring for example to Figure 2, the unexecuted BIOS may be made up of BIOS module 52 stored in the ROM 50, BIOS module 40 stored in the storage 38 associated with the server 36 and BIOS module contained on a smart card 48 read by a card reader 46. *See* Specification, p. 5.

The control of the various modules may be undertaken by an interpreter 52a which, in one embodiment of the present invention, may be stored on the BIOS ROM 50. The interpreter 52a contains all the directions to the various BIOS modules. It reads the various instructions and provides instructions about what to do under various circumstances. Thus, the interpreter 52a may be responsible for conditionally integrating the BIOS functions into an overall BIOS operation.

One example of an application of a distributed BIOS system in accordance with one embodiment of the present invention involves preboot authentication services (PAS). PAS provides different levels of authentication depending on the status of the processor-based system. For example, if the processor-based system 10 is connected to the network 34, the possibility of compromising system resources is much higher than if the system 10 were operated on a stand-alone basis without connection to network 34 resources. Therefore, in order to provide protection for those network resources, the authentication protocols for a new user may be implemented variably depending on the system state--in this case whether or not the system 10 is connected to a network 34. *See* Specification, pp. 5-6.

In PAS, when the system 10 is linked to the network 34, for example through a local adapter or NIC 32, this connection may be recognized, and the BIOS may control the nature of the authentication. Once network resources may be accessed, the authentication requirements may be increased. For example, there may be a token file which is also part of BIOS which may be placed on a network storage 38 such as a network drive rather than on the local system storage such as the BIOS ROM 50. The BIOS module 40 may be requested by the local system 10, loaded and executed for use in authentication.

A preboot execution environment (PXE) allows for remote new system startup. PXE provides a uniform protocol for the client to request the allocation of a network address and subsequently address the download of an NBP from a network boot server. It also provides a set of application program interfaces (APIs) available on the client's preboot firmware environment that constitute a set of services that can be employed by the NBP or the BIOS. Finally, PXE provides a standard method of initiating the preboot firmware to execute the PXE protocol on the client. A newly installed network client may enter a heterogeneous network, acquire a network address from a DHCP server and then download an NBP to set itself up. Thus, with PXE, a client such as the system 10 may be connected to a network in an automatic fashion, in the boot sequence, enabling the acquisition of a BIOS module from a remote network 34 during the boot process. *See* Specification, pp. 7-9.

Turning now to Figure 3, the BIOS may begin with the initialization of the processor 12, the memory, any chipsets and the like as indicated in block 54. This is the beginning of the boot process. Next, a connection to a network 34 may be detected as indicated in block 56. Next, as shown in block 58, the user authentication data may be captured and stored. Of course, the nature of the user authentication data may differ depending on whether the system is connected to a network. Thus, authentication data may come over the network 34 or from the storage 50 depending on the state of the system 10. After the user has been authenticated, a key may be obtained from protected storage. The OS is loaded, partitions are checked, and integrity is determined as indicated in block 62. In the post-operating system or post-OS space, the user may again be authenticated using operating system protocols as indicated in block 64. If encryption is built into the operating system, setup keys for encryption and decryption may be implemented in some embodiments as indicated in block 66.

Referring to Figure 4, in accordance with one embodiment of the present invention, a distributed BIOS file system 52 may be executed by determining what system state exists as indicated in diamond 72. Again, this information may be obtained using the PXE protocol. Depending on the system state (which in one embodiment of the present invention may be whether or not the system is connected to a network), a first BIOS module may be loaded as indicated in block 74 and executed as indicated in block 76. If a different system state exists, a second module may be loaded as indicated in block 78. Thereafter, this module may then be executed as indicated in block 80.

Thus, depending on the system state, either a first or a second module may be located and loaded. Those modules may or may not be distributed with respect to one another in that one of the modules, before execution, may be located in a storage different from the other of the modules. *See* Specification, pp. 9-10.

In some embodiments of the present invention, the BIOS may be broken down on logically self-encapsulated modules. The execution and the presence of each module is optional. A particular module may be present (or not) depending on system state, and the needs of a particular system. Similarly, a particular module may be executed or it may be skipped depending on the system policy, user choice or what is actually present on a particular system. *See* Specification, p. 11.

In some embodiments of the present invention, the distributed BIOS modules may be linked dynamically. A loader may be utilized to load the needed executable BIOS modules wherever located.

Referring to Figure 6, a given functional module 108 may have a predefined structure in accordance with one embodiment of the present invention. The module 108 may include a header 98 and the header 98 may include an entry point 100 which provides directions to locate one or more functions enabled by the module 108. Thus, the entry point 100 may include a direction 102 to code 104 for a first function implemented by a module and a direction 106 which directs the flow to a code 107 for a second function defined within the module 108. Thus, once the entry point 100 to the module 108 is located, its functions may be exposed.

The entry points for each module, as an offset from the module's base address, are generally stored in other modules. Thus, if a first module, that stores an offset to an entry point for a second module, finds the base memory address for the second module, the first module may link to the second module, in one embodiment of the present invention. A base memory address is defined by a shared descriptor table 100, shown in Figure 7. The descriptor table 100 stores information 112 about the base address 114 for one or more modules 108. For example, the descriptor table 100 may provide information 112 sufficient to locate the base address 114 of one or more modules 108. Thus, the descriptor table 100 may have information 112a which locates the base address 114a of the module 108a. *See* Specification, pp. 12-13.

Once the base address is located, the entry point can be determined from the offset. The offset provides the information about where the entry point is located in the module relative to the base address of the module. Once the entry point is determined, any function within the module may then be located. Each module may export its entry point. It may do this by providing information about the structure of its header, naming conventions, length of the header, and check sums and finally an offset from a base address to the entry point within the header. At run time, the segment address where the module was stored (and in particular its base address) is known. The segment address points to the location of the base address. The segment address information 112 may then be stored in the descriptor table 100. *See* Specification, pp. 12-14.

Referring next to Figure 8, the flow 110 for linking BIOS modules begins by compiling a BIOS module 108, as indicated in block 112. A header is then attached to the functional BIOS module as indicated in block 114. The header provides the function number that is used to locate particular functions implemented by the module. The entry point information is exported (block 116) from one module to other modules.

At run time, the system processor finds an appropriate free memory space to store each module, as indicated in block 118. The information 112 to locate the base address of the module is prepared (block 120), as indicated in block 120, and the functional BIOS module 108 is loaded into the free memory space at the base address (block 122). Thereafter, the module may be called by another module as indicated in block 124.

A module is called by accessing its segment address from the descriptor table 100. This information, together with the entry point, produces the information needed to locate each function, by function numbers, supported by the called module. In this way, one or more functional BIOS modules may be called by any other BIOS module to implement features which may or may not be desired in any particular system at any given time. *See* Specification, pp. 14-15.

VI. ISSUES

A. **Are Claims 1, 4, 5, 10, 22-23, and 30 Patentable Under 35 U.S.C. §102(e) Over Rakavy?**

B. **Are Claims 2, 11, and 13-17 Patentable Under 35 U.S.C. §102(e) Over Rakavy?**

C. **Are Claims 3, 12, and 24-27 Patentable Under 35 U.S.C. §102(e) Over Rakavy?**

D. **Are Claims 6, 7-9, 18-21, and 28-29 Patentable Under 35 U.S.C. §102(e) Over Rakavy?**

VII. GROUPING OF THE CLAIMS

For purposes of this appeal, the claims do not stand or fall together. Instead, for purposes of this appeal, Applicant has grouped together claims 1, 4, 5, 10, 22-23, and 30; claims 2, 11, and 13-17; claims 3, 12, and 24-27; and claims 6, 7-9, 18-21, and 28-29, as set forth above.

VIII. ARGUMENT

A. Claims 1, 4, 5, 10, 22-23, and 30 Are Patentable Under 35 U.S.C. §102(e) Over Rakavy

The method of claim 1 includes, *inter alia*, selectively loading either a first module of a basic input/output system (BIOS) or a second module of the BIOS based on a system state that indicates a connection to a network. Claim 1 stands rejected under 35 U.S.C. § 102(e) over U.S. Patent No. 6,324,644 to Rakavy, et al. (hereinafter, "Rakavy"). This rejection is improper.

In contrast to the method of claim 1, Rakavy teaches two different BIOS, one of which detects and loads the other regardless of a given system state. Therefore, Rakavy fails to disclose selective loading of one of two modules of BIOS based on a system state that indicates a connection to a network.

In Rakavy, a conventional BIOS 500 (hereinafter "conventional BIOS") detects and loads a network enhanced BIOS 600 (hereinafter "network enhanced BIOS"). In this way, one type of BIOS detects and loads another type of BIOS, instead of a first of two different modules of a single BIOS that is selectively loaded based on a system state that indicates a connection to a network.

That is, as described in column 6, lines 24-43 of Rakavy (contended by the Examiner to meet this element, *see* Final Office Action, p. 3), the conventional BIOS detects and loads the network enhanced BIOS to give control to the network enhanced BIOS. However, Rakavy does not disclose a system state that indicates a connection to network, upon which state the network enhanced BIOS is selectively loaded. Instead, enabler code 560 of the conventional BIOS merely detects and loads the network enhanced BIOS without regard to a system state, and certainly not such a state that is indicative of a network connection. Such operation is further detailed in FIG. 4A and FIG. 4B and the accompanying text in Rakavy. Rakavy, col. 7, lns. 4-32.

In the Advisory Action, the Examiner erroneously contends that columns 7 and 8 of Rakavy somehow disclose the use of a system state indicative of a network connection. *See* Advisory Action mailed April 9, 2004, p. 2. However, this discussion of state in Rakavy merely refers to actions occurring <u>after</u> a transition from the conventional BIOS to the network enhanced BIOS. As such, this disclosure in no way meets the claimed selective loading <u>based</u> on system state. For at least these reasons claim 1 and claims 4 and 5 depending therefrom are patentable over Rakavy. For the same reasons, claims 10 and 22 and claims 23 and 30 depending therefrom, are similarly patentable, and the rejection should be reversed.

B. Claims 2, 11, and 13-17 Are Patentable Under 35 U.S.C. §102(e) Over Rakavy

Claim 2 depends from claim 1 and further recites, *inter alia*, storing the first BIOS module in a first storage device prior to execution and storing a second BIOS module in a second storage device prior to execution. Claim 2 stands rejected under §102(e) over Rakavy. This rejection is improper, at least for the reasons discussed above regarding claim 1 (*see* VIII.A).

This rejection is further improper, as nowhere does Rakavy disclose storing its conventional BIOS and its network enhanced BIOS in two different storage devices. Instead, Rakavy merely discloses that the conventional BIOS is stored in a conventional manner, namely in a read only memory (ROM), i.e., flash ROM 125. *See* Rakavy, col. 2, lns. 5-14. Furthermore, Rakavy discloses that the network enhanced BIOS is similarly stored in flash ROM 125. Rakavy, col. 7, lns. 11-16, 22-25.

Accordingly, for this further reason, claim 2 and claims 11 and 13-17 are further patentable and the rejection should be reversed.

C. Claims 3, 12, and 24-27 Are Patentable Under 35 U.S.C. §102(e) Over Rakavy

Claim 3 depends from claim 2 and further recites that the second module is stored in a storage associated with a network server accessible to the system over a network. Claim 3 stands rejected under §102(e) over Rakavy. This rejection is improper, at least for the reasons discussed above with regard to claims 1 and 2, from which claim 3 depends (*see* VIII.A and VIII.B).

Claim 3 is further patentable, as nowhere does Rakavy disclose that the network enhanced BIOS is stored anywhere other than on desktop computer 400. That is, as described above, network enhanced BIOS is stored in flash ROM 125 of desktop computer 400. Accordingly, for this further reason, claims 3, 12, and 24-27 are further patentable, and the rejection should be reversed.

D. Claims 6, 7-9, 18-21, and 28-29 Are Patentable Under 35 U.S.C. §102(e) Over Rakavy

Claim 6 depends from claim 1 and further recites dynamically linking to one of a plurality of modules, and exporting an offset to an entry point in one module to another module. Claim 6 stands rejected under §102(e) over Rakavy. This rejection is improper, at least for the reasons discussed above regarding claim 1 (*see* VIII.A).

The rejection is further improper, as Rakavy does not disclose exporting an offset to an entry point in one module to another module. In this regard, the portion of Rakavy contended to meet this element (*see* Rakavy, col. 7, lns. 25-33 and col. 8, lns. 1-6) nowhere discloses that an offset to an entry point in one module is exported to another module. Accordingly, for this further reason, claim 6 and claims 7-9 depending therefrom are patentable over Rakavy. For similar reasons, claims 18-21 and 28 and 29 are patentable over Rakavy, and the rejection should be reversed.

IX. CONCLUSION

Since the rejections of the claims are baseless, they should be reversed.

Respectfully submitted,

Date: 2/2/04

Mark J. Rozman
Registration No. 42,117
TROP, PRUNER & HU, P.C.
8554 Katy Fwy, Ste 100
Houston, TX 77024-1805
512/418-9944 [Phone]
713/468-8883 [Facsimile]

21906
PATENT TRADEMARK OFFICE

APPENDIX OF CLAIMS

The claims on appeal are:

1. A method comprising:

selectively loading either a first module of the basic input/output system or a second module of the basic input/output system based on a system state that indicates a connection to a network;

executing said first basic input/output system module; and

dynamically linking to said second basic input/output system module.

2. The method of claim 1 further comprising:

storing said first module of a basic input/output system for a processor-based system on a first storage device prior to execution;

storing said second module of the basic input/output system on a second storage device prior to execution; and

enabling said second module to be executed conditionally depending on a state of said processor-based system.

3. The method of claim 2 wherein storing said second module includes storing said second module in a storage associated with a network server accessible to said processor-based system over a network.

4. The method of claim 1 further including detecting said system state during the boot sequence.

5. The method of claim 4 including detecting whether or not the system is connected to a network during the boot operation.

6. The method of claim 1 including dynamically linking to one of a plurality of modules, and exporting an offset to an entry point in one module to another module.

7. The method of claim 6 including storing a secondary entry point in a module to locate a function within the module.

8. The method of claim 7 including developing a segment address for said second module at run time.

9. The method of claim 8 including providing a descriptor table which indicates a segment address for said second module.

10. An article comprising a medium for storing instructions that cause a processor-based system to:

selectively load either a first module of the basic input/output system or a second module of the basic input/output system based on a system state that indicates a connection to a network;

execute said first basic input/output system module; and

dynamically link to said second basic input/output system module.

11. The article of claim 10 further storing instructions that cause a processor-based system to:

access said first module of a basic input/output system on a first storage device;

access said second module of the basic input/output system on a second storage device; and

execute said second module conditionally depending on the state of said processor-based system.

12. The article of claim 11 further storing instructions that cause a processor-based system to access said second module in a storage associated with a network server accessible to said processor-based system over a network.

13. The article of claim 11 further storing instructions that cause a processor-based system to execute said second module conditionally depending on whether or not the processor-based system is coupled to a network.

14. The article of claim 11 further storing instructions that cause a processor-based system to selectively access either a first module setting forth a first authentication protocol in a first storage device or a second module setting forth a second authentication protocol in a second storage device.

15. The article of claim 11 further storing instructions that cause a processor-based system to dynamically link said first and second modules.

16. The article of claim 11 further storing instructions that cause a processor-based system to detect said system state during the boot sequence.

17. The article of claim 16 further storing instructions that cause a processor-based system to detect whether the system is connected to a network during the boot operation.

18. The article of claim 11 further storing instructions that cause a processor-based system to dynamically link to one of a plurality of modules using offsets to entry points in said modules.

19. The article of claim 18 further storing instructions that cause a processor-based system to store a secondary entry point in a module to locate a function within the module.

20. The article of claim 19 further storing instructions that cause a processor-based system to develop a segment address for said second module at run time.

21. The article of claim 20 further storing instructions that cause a processor-based system to provide a descriptor table which identifies the segment address for said second module.

22. A processor-based system comprising:

a processor;

a first basic input/output system module executable by said processor; and

a second basic input/output system module executable by said processor, said second module being dynamically linked to said first module after selectively loading either said first module of the basic input/output system or said second module of the basic input/output system based on a system state that indicates a connection to a network.

23. The system of claim 22 including a detector that detects a system state to determine whether said processor executes said second module.

24. The system of claim 22 including a first storage for said first module and a second storage for said second module, said second storage being coupled to said processor-based system over a network.

25. The system of claim 24 wherein said detector detects information about network access.

26. The system of claim 25 wherein said first and second modules include different authentication protocols.

27. The system of claim 26 wherein said processor executes said basic input/output system module on said second storage to implement a network authentication protocol.

28. The system of claim 22 wherein said first module dynamically links to said second module, using an offset exported from said second module.

29. The system of claim 28 wherein said first module uses a secondary entry point to locate a function in said second module.

30. The system of claim 22 wherein said processor provides a descriptor table which includes a segment address for said second module.

AF/2135/#20

TRANSMITTAL OF APPEAL BRIEF (Large Entity)

Docket No.
ITL.0304US

In Re Application Of: **Alex I. Eydelberg**

Serial No.	Filing Date	Examiner	Group Art Unit
09/465,600	**December 17, 1999**	**Leynna A. Ha**	**2135**

Invention: **Dynamically Linked Basic Input/Output System**

TO THE COMMISSIONER FOR PATENTS:

Transmitted herewith in triplicate is the Appeal Brief in this application, with respect to the Notice of Appeal filed on **May 7, 2004**

The fee for filing this Appeal Brief is: **$330.00**

☒ A check in the amount of the fee is enclosed.

□ The Director has already been authorized to charge fees in this application to a Deposit Account.

☒ The Director is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. **20-1504**

Mark Rozman

Signature

Dated: **July 7, 2004**

Mark J. Rozman, Reg. No. 42,117
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, TX 77024

21906
PATENT TRADEMARK OFFICE

I certify that this document and fee is being deposited on **July 7, 2004** with the U.S. Postal Service as first class mail under 37 C.F.R. 1.8 and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Signature of Person Mailing Correspondence

Jennifer Juarez

Typed or Printed Name of Person Mailing Correspondence

cc: